

UNCONDITIONAL CLASS GROUP TABULATION OF IMAGINARY QUADRATIC FIELDS TO $|\Delta| < 2^{40}$

A. S. MOSUNOV AND M. J. JACOBSON, JR.

ABSTRACT. We present an improved algorithm for tabulating class groups of imaginary quadratic fields of bounded discriminant. Our method uses classical class number formulas involving theta-series to compute the group orders unconditionally for all $\Delta \not\equiv 1 \pmod{8}$. The group structure is resolved using the factorization of the group order. The $1 \pmod{8}$ case was handled using the methods of [JRW06], including the batch verification method based on the Eichler-Selberg trace formula to remove dependence on the Extended Riemann Hypothesis. Our new method enabled us to extend the previous bound of $|\Delta| < 2 \cdot 10^{11}$ to 2^{40} . Statistical data in support of a variety conjectures is presented, along with new examples of class groups with exotic structures.

1. INTRODUCTION

The class group of an imaginary quadratic field $\mathbb{Q}(\sqrt{\Delta})$ with discriminant Δ , denoted by Cl_{Δ} , has been studied extensively over the past two centuries. Many things are known about the class group. For example, if we know the *class number* $h(\Delta)$, which is defined as the size of Cl_{Δ} , we can find a non-trivial factor of Δ . Also, from the prime factorization of Δ we can determine the parity of $h(\Delta)$, as well as the rank of the 2-Sylow subgroup of Cl_{Δ} .

However, the number of open questions about Cl_{Δ} most certainly exceeds the number of answered. For example, computing the class number is believed to be computationally difficult; it is known to be at least as hard as integer factorization, and is currently harder. The heuristics of Cohen and Lenstra [CL84] allow us to make certain predictions regarding divisibility properties of $h(\Delta)$ and the structure of Cl_{Δ} , but most of these, especially with respect to odd primes, remain unproved. Another question of interest is to provide tight bounds on $h(\Delta)$. This has been answered by Littlewood [Lit28], but the result is *conditional*; that is, it depends on the *Extended Riemann Hypothesis* (ERH).

Due to the lack of unconditional proof on such basic arithmetic properties on Cl_{Δ} , it is of interest to provide numerical evidence supporting the heuristics and conditional results. Tabulating Cl_{Δ} for as many small discriminants as possible provides such evidence. The first major work on class group tabulation is due to Buell, who in a series of papers culminating in [Bue99], computed all Cl_{Δ} for negative Δ satisfying $|\Delta| < 2.2 \cdot 10^9$. In his work, Buell gathered statistics on

2010 *Mathematics Subject Classification.* Primary 11R29; Secondary 11R11, 11Y40.

Key words and phrases. binary quadratic form, class group, tabulation, out-of-core multiplication, Cohen-Lenstra heuristics.

The first author's research was supported by "Alberta Innovates Technology Futures", Canada. The second author's research is supported by NSERC of Canada.

Littlewood’s bounds on $L(1, \chi_\Delta)$ [Lit28], Bach’s bound on the size of the generators required to produce Cl_Δ [Bac90], and the Cohen-Lenstra heuristics [CL84]. He also provided a table of first occurrences of so-called “exotic” groups. These groups possess interesting group structures, such as non-cyclic p -Sylow subgroups for odd primes p , which according to the Cohen-Lenstra heuristics are quite rare. Such groups are of interest, for example, in the context of class field theory, as the towers of field extensions for them have interesting, non-trivial properties [Mey12].

The next and also most recent work of interest is due to Jacobson et al. [JRW06], who used a baby-step giant-step algorithm to tabulate all class groups to 10^{11} [BJT97, Algorithm 4.1]. The bound was further extended to $2 \cdot 10^{11}$ in the Master’s thesis of Ramachandran [Ram06]. The authors used Bach’s averaging method in order to determine a *conditional* lower bound h^* on $h(\Delta)$, such that $h^* \leq h(\Delta) \leq 2h^*$ [Bac95]. Due to the nature of the baby-step giant-step algorithm, knowing this bound was sufficient to be certain that the whole group was generated, assuming the ERH. In order to eliminate the ERH dependency, they applied the Eichler-Selberg trace formula [SvdV91], which relates sums of Hurwitz class numbers to the trace of a certain Hecke operator [JRW06, Formula 2.2]. Following Buell, the authors gathered statistics on various hypotheses regarding Cl_Δ .

In this paper, we push the feasibility limit further by tabulating class groups for all negative Δ such that $|\Delta| < 2^{40} = 1.09951 \dots \times 10^{12}$. Using certain class number generating functions [Wat35], we were able to compute all class numbers $h(\Delta)$ for $\Delta \not\equiv 1 \pmod{8}$ via a product of two large-degree power series; this method was inspired by that of Hart et al. to tabulate all congruent numbers to 10^{12} [HTW10]. Computing the class numbers first allowed us to further achieve a significant speedup in class group tabulation for these Δ by only resolving the structure of possibly non-cyclic subgroups, i.e., for which all prime divisors of the order occur with multiplicity greater than one. The discriminants $\Delta \equiv 1 \pmod{8}$ were handled separately using the previous technique of Jacobson et al. [JRW06]. In the end, we observed that the class groups with $\Delta \not\equiv 1 \pmod{8}$ were computed over 4.72 times faster than class groups with $\Delta \equiv 1 \pmod{8}$.

Unfortunately, the improved running time did not come for free, as we were no longer able to test Bach’s bound on the size of generators required to produce the whole group [Bac90]. Nevertheless, we were still able to gather extensive computational evidence in support of Littlewood’s bounds [Lit28], the Cohen-Lenstra heuristics [CL84], and Euler’s hypothesis on idoneal numbers [Kan11]. We also further extended Buell’s table of exotic groups [Bue99].

Our paper is organized as follows. In Section 3, we present three formulas suitable for the tabulation of class numbers $h(\Delta)$ with $\Delta \not\equiv 1 \pmod{8}$. Section 4 is dedicated to the out-of-core polynomial multiplication technique due to Hart et al. [HTW10], which allows to compute the product of two large polynomials that cannot fit into memory all together. Section 5 gives a brief overview of the techniques that were used in order to tabulate Cl_Δ for $\Delta \equiv 1 \pmod{8}$. Section 6 discusses the performance of our program. In Section 7, we present our numerical results, which include statistics on various hypotheses regarding Cl_Δ and the refined table of exotic groups. Section 8 concludes the paper by giving a discussion of various techniques, which can further accelerate the class number and class group tabulation.

2. PRELIMINARIES

Our method for computing the class numbers relies on classical results related to binary quadratic forms. Hence, our algorithms will be described in the language of forms, and we will use the fact that the ideal class group of the field $\mathbb{Q}(\sqrt{\Delta})$ of discriminant $\Delta < 0$ is isomorphic to the group of equivalence classes of binary quadratic forms of discriminant Δ .

In particular, we consider binary quadratic forms from two different perspectives. We use (a, b, c) to denote a *modern* binary quadratic form, i.e. form which possesses a *discriminant* $\Delta = b^2 - 4ac$. We also use $(a, 2b, c)$ to denote a *classical* binary quadratic form of *determinant* $D = b^2 - ac$, studied by Gauß [Gau86, Chapter 5]. In the first case, the set of equivalence classes with respect to invertible integral linear changes of variables forms a group under composition of forms. An analogous observation can be made regarding the set of *properly primitive* (to be defined) classical quadratic forms. In the first case, the class group of modern forms is isomorphic to the ideal class group of $\mathbb{Q}(\sqrt{\Delta})$ whenever Δ is a field discriminant (square-free integer congruent to 1 mod 4 or 4 times a square-free number). The second case is closely related; as the formula (3.4) suggests, the resulting group order corresponding to determinant D differs from $h(\Delta)$ by a factor of three if $\Delta \equiv 5 \pmod{8}$, $\Delta \neq -3$ and is equal to $h(\Delta)$ otherwise.

We also require the following classifications of classical forms:

Definition 2.1. [Gau86, §226] Consider a quadratic form $(a, 2b, c)$ and its *divisor* $\delta = \gcd(a, b, c)$. Then $(a, 2b, c)$ is called *primitive* if $\delta = 1$, and *derived* otherwise.

Definition 2.2. [Kro60] A primitive quadratic form $(a, 2b, c)$ is called *uneven* if $\gcd(a, 2b, c) = 1$, i.e., its coefficients a and c are not both even; it is called *even* otherwise. A derived form of divisor δ is uneven when $(a/\delta, 2b/\delta, c/\delta)$ is uneven; otherwise it is even.

By $F(n)$ and $F_1(n)$ Kronecker denoted the total number of uneven and even equivalence classes of forms of determinant $D = -n$, respectively. We extend his notation by writing $\tilde{F}(n)$ and $\tilde{F}_1(n)$ for the total number of *primitive* uneven and even equivalence classes of determinant $D = -n$, respectively. Note that there exists a straightforward connection between $F(n)$ and $\tilde{F}(n)$, and between $F_1(n)$ and $\tilde{F}_1(n)$. In particular, if we write $n = g^2e$, where e is square-free, then

$$(2.1) \quad F(n) = \sum_{t|g} \tilde{F}\left(\frac{n}{t^2}\right) \quad \text{and} \quad F_1(n) = \sum_{t|g} \tilde{F}_1\left(\frac{n}{t^2}\right).$$

To see this, observe that when $\gcd(a, b, c) = t > 1$, from every uneven primitive form $(a/t, 2b/t, c/t)$ of determinant $-n/t^2$ we can obtain every uneven derived form $(a, 2b, c)$ of determinant $D = -n$ and divisor $\delta = t$. By counting all uneven primitive forms with all uneven derived ones, we obtain $F(n)$. A similar reasoning allows us to deduce the formula for $F_1(n)$.

3. CLASS NUMBER TABULATION FORMULAS

We begin by considering the following Jacobi theta series:

$$\vartheta_2(q) = 2 \sum_{k=0}^{\infty} q^{\left(k+\frac{1}{2}\right)^2} = 2q^{\frac{1}{4}} + 2q^{\frac{9}{4}} + 2q^{\frac{25}{4}} + 2q^{\frac{49}{4}} + \dots ;$$

$$\vartheta_3(q) = 1 + 2 \sum_{k=1}^{\infty} q^{k^2} = 1 + 2q + 2q^4 + 2q^9 + 2q^{16} + \dots$$

In 1860, Kronecker found the connection that exists between $\vartheta_3(q)$ and classical quadratic forms. We summarize his result in Theorem 3.1.

Theorem 3.1 ([Kro60]). *Let $F(n)$ and $F_1(n)$ count equivalence classes $[(1, 2 \cdot 0, 1)]$ and $[(2, 2 \cdot 1, 2)]$, and classes derived from them, as $1/2$ and $1/3$, respectively. Define $E(0) = 1/12$, $E(4n) = E(n)$ for $n \neq 0$, and $E(n) = F(n) - F_1(n)$ for $n \not\equiv 0 \pmod{4}$. Then*

$$(3.2) \quad \vartheta_3^3(q) = 12 \sum_{n=0}^{\infty} E(n)q^n.$$

Though not obvious at first sight, the formula (3.2) allows us to tabulate class numbers $h(\Delta)$. Recall Gauß's result that $\tilde{F}(n)$ is a multiple of $\tilde{F}_1(n)$ [Gau86, §256]:

$$(3.3) \quad \tilde{F}_1(n) = \begin{cases} \tilde{F}(n), & \text{when } n \equiv 7 \pmod{8} \text{ or } n = 3; \\ \tilde{F}(n)/3, & \text{when } n \equiv 3 \pmod{8} \text{ and } n \neq 3; \\ 0, & \text{when } n \not\equiv 3 \pmod{4}. \end{cases}$$

We may now prove Theorem 3.2, which connects $h(\Delta)$ to $\tilde{F}(n)$, where $\Delta = -4n$ or $\Delta = -n$, depending on the congruence class of Δ modulo 4.

Theorem 3.2. *For $\Delta < 0$ the following relation holds:*

$$(3.4) \quad h(\Delta) = \begin{cases} \tilde{F}(n), & \text{when } \Delta \equiv 0, 1, 4 \pmod{8} \text{ or } \Delta = -3; \\ \tilde{F}(n)/3, & \text{when } \Delta \equiv 5 \pmod{8} \text{ and } \Delta \neq -3, \end{cases}$$

where

$$(3.5) \quad n = \begin{cases} -\Delta/4, & \text{if } \Delta \equiv 0 \pmod{4}; \\ -\Delta, & \text{if } \Delta \equiv 1 \pmod{4}. \end{cases}$$

Proof. Consider a primitive binary quadratic form $(a, 2b, c)$ of determinant $D = -n$, where n is positive. When $D \equiv 1 \pmod{4}$ and $(a, 2b, c)$ is even, i.e. a, c are even and b is odd, this form can be transformed into a form $(a/2, b, c/2)$ with discriminant $\Delta \equiv 1 \pmod{4}$. This map is bijective, since every form (a, b, c) of discriminant Δ with odd b corresponds to a primitive even form $(2a, 2b, 2c)$ of determinant D . We conclude that $h(\Delta) = \tilde{F}_1(n)$. When $D \not\equiv 1 \pmod{4}$, there are no primitive even forms, and a primitive uneven form $(a, 2b, c)$ with determinant D already has a fundamental discriminant $\Delta = 4D$. There are no other forms (a, b, c) of discriminant Δ with $\gcd(a, b, c) = 1$, besides those of determinant D , so $h(\Delta) = \tilde{F}(n)$. In the end, we obtain the relation (3.4). \square

According to Theorems 3.1 and 3.2, by cubing $\vartheta_3(q)$ we can tabulate $h(\Delta)$ for every fundamental discriminant Δ , except for $\Delta \equiv 1 \pmod{8}$, because in this case we have $F(n) = F_1(n)$ and thus $E(n) = 0$.

Before proceeding further, recall the definition of a Hurwitz class number $H(n)$.

Definition 3.3. Let

$$(3.6) \quad h_\omega(\Delta) = \begin{cases} h(\Delta), & \text{if } \Delta < -4; \\ 1/2, & \text{if } \Delta = -4; \\ 1/3, & \text{if } \Delta = -3, \end{cases}$$

and consider negative $\Delta = f^2\Delta_1$, where Δ_1 is a fundamental discriminant. Then

$$H(|\Delta|) = \sum_{t|f} h_\omega\left(\frac{\Delta}{t^2}\right),$$

is called the *Hurwitz class number*.

In Theorem 3.2 we determined the connection that exists between $h(\Delta)$ and the number of *primitive* uneven classes $\tilde{F}(n)$. However, the formula (3.2) has $F(n)$ instead of $\tilde{F}(n)$, which also take the *derived* uneven classes into account. In fact, it is not hard to prove that $F(n) = \tilde{F}(n)$ and $F_1(n) = \tilde{F}_1(n)$ hold if and only if n is square-free. In order to establish this connection for an arbitrary n , we aim to prove Theorem 3.4, which relates $F(n)$ to the Hurwitz class number $H(n)$ or $H(4n)$, depending on the congruence class of n modulo 4. To the best of our knowledge, the formula (3.7) is not present in any literature available, though its statement for the special case of square-free $n > 4$ is well known and can be found, for example, in the monograph of Grosswald [Gro85, Chapter 4, Theorem 2].

Theorem 3.4. *Let $E(n)$ be as in Theorem 3.1. Then*

$$(3.7) \quad E(n) = \begin{cases} 1/12, & \text{when } n = 0; \\ E(n/4), & \text{when } n \equiv 0 \pmod{4} \text{ and } n \neq 0; \\ H(4n), & \text{when } n \equiv 1, 2 \pmod{4}; \\ 2H(n), & \text{when } n \equiv 3 \pmod{8}; \\ 0, & \text{when } n \equiv 7 \pmod{8}, \end{cases}$$

where $H(n)$ denotes the Hurwitz class number.

Proof. Consider the following two cases, corresponding to square-free values of n :

- (1) Let $n \equiv 1, 2 \pmod{4}$. For $n = 1$, we can verify that the formula (3.7) gives us the correct result. For $n \neq 1$, from (3.3) we know that $\tilde{F}_1(n) = 0$, and from (3.4) we know that $h(\Delta) = \tilde{F}(n)$, where $\Delta = -4n$ according to the relation (3.5). Therefore, $E(n) = \tilde{F}(n) - \tilde{F}_1(n) = h_\omega(-4n)$;
- (2) Let $n \equiv 3 \pmod{8}$. For $n = 3$, we can verify that the formula (3.7) gives us the correct result. For $n \neq 3$, from (3.3) we know that $\tilde{F}(n) = 3\tilde{F}_1(n)$, and from (3.4) we know that $h(\Delta) = \tilde{F}_1(n)$, where $\Delta = -n$ according to the relation (3.5). We obtain $E(n) = \tilde{F}(n) - \tilde{F}_1(n) = 2h_\omega(-n)$.

Now, consider an arbitrary $n = g^2e$, where e is square-free. If we now recall formulas from (2.1), then by Definition 3.3 we obtain formulas for $n \equiv 1, 2 \pmod{4}$ and $n \equiv 3 \pmod{8}$:

$$\begin{aligned} E(n) &= \sum_{t|g} \left[\tilde{F}\left(\frac{n}{t^2}\right) - \tilde{F}_1\left(\frac{n}{t^2}\right) \right] \\ &= \begin{cases} \sum_{t|g} h_\omega\left(\frac{-4n}{t^2}\right) = H(4n), & \text{if } n \equiv 1, 2 \pmod{4}; \\ 2 \sum_{t|g} h_\omega\left(\frac{-n}{t^2}\right) = 2H(n), & \text{if } n \equiv 3 \pmod{8}. \end{cases} \end{aligned}$$

According to Theorem 3.4, by cubing $\vartheta_3(q)$ we can tabulate Hurwitz class numbers $H(n)$. However, the formula (3.2) is quite inefficient for our purposes, as we are interested in only fundamental discriminants Δ , which correspond to coefficients of

$\vartheta_3^3(q)$ of the form $16k + 4$, $16k + 8$, $8k + 3$ and $8k + 7$ for some non-negative integer k . We expect to have around $(3/\pi^2)N \approx 0.304N$ fundamental discriminants, satisfying $|\Delta| < N$ [Coh93, Section 5.10]. Fortunately, there exist three alternative formulas, namely (1.13), (1.12) and (1.14) of [Wat35], which can be derived easily from (3.2) [Bel24]:

$$(3.8) \quad \sum_{k=0}^{\infty} F(4k+2)q^k = \nabla^2(q^2)\vartheta_3(q);$$

$$(3.9) \quad 2 \sum_{k=0}^{\infty} F(4k+1)q^k = \nabla(q^2)\vartheta_3^2(q);$$

$$(3.10) \quad \sum_{k=0}^{\infty} F(8k+3)q^k = \nabla^3(q),$$

where

$$\begin{aligned} \nabla(q) &= \frac{1}{2} \vartheta_2(\sqrt{q})q^{-\frac{1}{8}} \\ &= \frac{1}{2} \cdot 2 \sum_{k=0}^{\infty} \sqrt{q}^{(k+\frac{1}{2})^2} \cdot q^{-\frac{1}{8}} \\ &= \sum_{k=0}^{\infty} q^{\frac{k(k+1)}{2}} = 1 + q + q^3 + q^6 + q^{10} + \dots \end{aligned}$$

In order to tabulate all class numbers corresponding to fundamental $\Delta \not\equiv 1 \pmod{8}$ and $|\Delta| < N$, it is sufficient to compute (3.8) and (3.9) to degrees $\lfloor N/16 \rfloor$, and (3.10) to degree $\lfloor N/8 \rfloor$. This can be done by multiplying polynomials, obtained by truncating series on the right sides of the equations above to a specific degree.

Although this idea of reducing the class number tabulation problem sounds good in theory, there are significant practical obstacles when large bounds on the discriminant are considered. In particular, the polynomials involved are too large to fit into computer memory, so we have to perform our multiplication out-of-core, i.e. with the usage of the hard disk. We discuss the out-of-core polynomial multiplication technique in Section 4.

4. OUT-OF-CORE MULTIPLICATION

In order to compute $h(\Delta)$ for all fundamental discriminants $\Delta < 0$, satisfying $|\Delta| < N$ and $\Delta \not\equiv 1 \pmod{8}$, we aim to compute relations (3.8), (3.9) and (3.10) to degrees $\lfloor N/16 \rfloor$, $\lfloor N/16 \rfloor$ and $\lfloor N/8 \rfloor$, respectively.

In our computations, N was chosen to be 2^{40} . If we assume that each coefficient of some polynomial $f(x)$ of degree $\lfloor N/8 \rfloor$ fits into 4 bytes, then we would require 512 GB to fit $f(x)$ into memory. Hence, in order to store two polynomials, $f(x)$ and $g(x)$, as well as the resulting polynomial $h(x)$, we need 1.5 TB, not to mention that the Fast-Fourier Transform (FFT), which we use to multiply polynomials, requires a lot of memory for intermediate results. Such an intensive memory requirement forces us to perform polynomial multiplication *out-of-core*, i.e., with the usage of the hard disk.

The first step is to reduce the degree of the polynomials to be multiplied. Following Hart et al. [HTW10], we convert polynomials of large degree with small

coefficients into polynomials of small degree with large coefficients by utilizing *Kronecker substitution*. Consider the polynomial

$$f(x) = f_0 + f_1x + f_2x^2 + \dots + f_{N-1}x^{N-1} \in \mathbb{Z}[x]$$

of degree $N - 1$. Fix a *bundling parameter* B , dividing N , and let $N_0 = N/B$. Then we can write $\hat{f}(x, y)$, satisfying $\hat{f}(x^B, x) = f(x)$, as follows:

$$\hat{f}(x, y) = \sum_{n=0}^{N_0-1} F_n(y)x^n = F_0(y) + F_1(y)x + F_2(y)x^2 + \dots + F_{N_0-1}(y)x^{N_0-1},$$

where

$$F_n(y) = f_{nB} + f_{nB+1}y + \dots + f_{(n+1)B-1}y^{B-1}.$$

If all the coefficients of $f(x)$ fit into s bits, we can *bundle* them by evaluating each $F_n(y)$ at 2^s , and obtain the following *bundled polynomial* $F(x)$:

$$(4.11) \quad F(x) = \sum_{n=0}^{N_0-1} F_n(2^s)x^n.$$

While $f(x)$ has coefficients of size s bits and degree $N - 1$, the bundled polynomial $F(x)$ has coefficients of size Bs bits and a smaller degree $N_0 - 1$. Now, in order to perform a multiplication $h(x) = f(x) \times g(x)$, one has to bundle coefficients of $g(x)$ with the same parameters B and s , and obtain a bundled polynomial $G(x)$. The polynomial $H(x) = F(x) \times G(x)$, the coefficients of which fit into $(2B - 1)s$ bits, will therefore embed information on coefficients of $h(x)$.

As a technical point, note that $H(x)$ is *not* a bundled polynomial of $h(x)$. In order to extract the coefficients of $h(x) = \sum_{k=0}^{N-1} h_k x^k$ from $H(x) = \sum_{n=0}^{N_0-1} H_n x^n$, a simple computation reveals that the summands of $h_k = \sum_{i=0}^k f_i g_{k-i}$ with $nB \leq k \leq nB + B - 2$ occur in both H_{n-1} and H_n for some positive integer n . In fact, if we let $H_n = \sum_{j=0}^{2B-2} H_n^{(j)} 2^{js}$, where $H_n^{(j)}$ are all positive, then $h_k = H_n^{(k)} + H_{n-1}^{(B+k)}$. The only exceptions correspond to $h_k = H_0^{(k)}$ for $k < B$, and for $h_{tB-1} = H_{t-1}^{(tB-1)}$ for some integer $t > 1$. Nevertheless, it is a simple matter to recover the h_k given $H(x)$.

At this point we have reduced the problem to a multiplication of smaller-degree polynomials, but with much larger coefficients. The next step is to reduce the coefficient sizes to the point that the polynomials involved can be fit into available memory. This is accomplished via many *Number Theoretic Transforms* (NTT) with *Chinese Remainder Theorem* (CRT) reconstitution¹. The idea is simple: in order to multiply two bundled polynomials $F(x)$ and $G(x)$ with large coefficients, one chooses n many primes p_0, \dots, p_{n-1} , and performs reduction of coefficients of $F(x)$ and $G(x)$ modulo each p_i for $0 \leq i < n$ using a *remainder tree* [BM74]. After that, n pairs of polynomials are multiplied (possibly in parallel) over each finite field \mathbb{F}_{p_i} , and as a result, each polynomial will contain residues of $H(x) = F(x) \times G(x)$ modulo p_i . In the end, the coefficients of $H(x)$ can be reconstructed with the Chinese Remainder Theorem, and this procedure can also be easily parallelized. Note that the intermediate results, namely reduced polynomials and the result of polynomial multiplications, are stored in m files on the hard disk. We observed that the choice of the number of files does not affect the performance of our program, and suggest

¹The paper of Hart contains a good survey on various out-of-core FFT methods and their applications [HTW10, Section 3].

to set m to be an integer multiple of number of threads used for computations. In our computations, we used 64 threads and produced $m = 2^{12} = 4096$ files for each congruence class of Δ . With this choice of m , each file contains at most 10.3 million class groups, providing a reasonable balance between file size and total number of files.

In order for the technique described previously to work one has to know ahead an upper bound C on coefficients of $h(x)$, and choose primes such that $C < \prod_{i=0}^{n-1} p_i$. Depending on the amount of memory available, each p_i is chosen in such a way that the reduced polynomials in $\mathbb{F}_{p_i}[x]$ can be comfortably multiplied in main memory.

4.1. Computational Parameters. The choices of a bundling parameter B and number of CRT primes can be optimized based on the amount of computer memory available. In order to make a proper choice of the *bit size parameter* s , we need to know how many bits are required to represent coefficients of (3.8), (3.9) and (3.10). In other words, we need to determine an explicit, unconditional upper bound on $H(n)$. To this end, consider the analytic class number formula

$$h_\omega(\Delta) = \frac{1}{\pi} \sqrt{|\Delta|} L(1, \chi_\Delta), \text{ where } L(1, \chi_\Delta) = \sum_{m=1}^{\infty} \frac{1}{m} \left(\frac{\Delta}{m} \right),$$

$h_\omega(\Delta)$ is defined in (3.6), and $\left(\frac{\Delta}{m} \right)$ is the Dirichlet symbol. To find the bit size of $h(\Delta)$, we utilize Ramaré's unconditional bound on $L(1, \chi_\Delta)$ [Ram01]:

$$(4.12) \quad L(1, \chi_\Delta) \leq a \log |\Delta| + b,$$

where

$$\begin{aligned} a &= \frac{1}{4}, b = \frac{5}{4} - \frac{\log 3}{2}, & \text{if } \Delta \equiv 0 \pmod{4}; \\ a &= \frac{1}{2}, b = \frac{5}{2} - \log 6, & \text{if } \Delta \equiv 1 \pmod{4}. \end{aligned}$$

We may now apply Ramaré's bounds (4.12) to determine an upper bound on $H(n)$ for every $n < N$ of the form $n = g^2 e$, where e is square-free:

$$H(n) \leq \frac{1}{\pi} \sum_{t|g} \sqrt{\frac{n}{t^2}} \left(a \log \frac{n}{t^2} + b \right) < \frac{1}{\pi} \sqrt{n} (a \log |\Delta| + b) \sum_{t|g} \frac{1}{t}.$$

To estimate the sum $\sum_{t|g} \frac{1}{t}$ for $N = 2^{40}$, we picked the largest possible $g = 605395$, and found the integer $n = 554400$ that does not exceed g , which has the largest value of $\sum_{t|n} \frac{1}{t} = \frac{1209}{275}$. Then, for

$$(4.13) \quad C_N = \left\lfloor \frac{1209}{275} \cdot \frac{1}{\pi} \sqrt{N} (a \log N + b) \right\rfloor, \text{ where } N \leq 2^{40},$$

and a and b as in (4.12), we have that $H(n) < C_N$ for all $n < N$.

Now we can explain how to compute the bit size parameter s . Recall that the main class number tabulation formulas (3.8), (3.9) and (3.10) require C_N , $2C_N$ and $3C_N$ as their upper bounds, respectively. Considering this, the formula for s is given as

$$(4.14) \quad s = \begin{cases} \lceil \log_2 C_N \rceil, & \text{for (3.8);} \\ \lceil \log_2 (2C_N) \rceil, & \text{for (3.9);} \\ \lceil \log_2 (3C_N) \rceil, & \text{for (3.10),} \end{cases} \text{ where } N \leq 2^{40}.$$

Finally, we need to determine how many primes to choose with respect to the bundling parameter B in order to restore coefficients of $H(x) = F(x) \times G(x)$, which

all fit into s bits. Recall that each coefficient of $H(x)$ has size $(2B - 1)s$. In order to restore coefficients of $H(x)$ with the CRT algorithm, we need to pick the primes p_0, \dots, p_{n-1} so that $(2B - 1)s < \log_2(p_0 \cdot \dots \cdot p_{n-1})$. In our implementation, we chose the smallest prime p_0 exceeding some positive lower bound P , and $n - 1$ primes p_1, \dots, p_{n-1} , which consecutively follow after p_0 . We choose n so that

$$(2B - 1)s \leq n \log_2 p_0 < \sum_{i=0}^{n-1} \log_2 p_i,$$

i.e.

$$(4.15) \quad n = \left\lceil \frac{(2B - 1)s}{\log_2 p_0} \right\rceil.$$

Note that for large p_0 and small n , the difference between $n \log_2 p_0$ and $\sum_{i=0}^{n-1} \log_2 p_i$ becomes negligible. Following Hart et al. [HTW10], we chose p_0 to be the smallest prime exceeding $P = 2^{62}$, which fits into a single machine word on a 64-bit system.

4.2. Complexity Analysis. Before proceeding to the complexity analysis, we first summarize the process of computation of $h(x) = f(x) \times g(x)$. Given two polynomials, $f(x)$ and $g(x)$, both of degree $N - 1$, the bundling parameter B (which for convenience divides N), the bit size parameter s , and n primes p_0, \dots, p_{n-1} , we compute the product of two polynomials in five stages:

- (1) Compute the bundled polynomials $F(x)$ and $G(x)$ of $f(x)$ and $g(x)$, respectively, using Kronecker substitution;
- (2) Reduce the coefficients of $F(x)$ and $G(x)$ modulo primes p_0, \dots, p_{n-1} using the remainder tree [BM74] in order to obtain the reduced polynomials $F_{p_i}(x)$ and $G_{p_i}(x)$ in $\mathbb{F}_{p_i}[x]$ for $0 \leq i < n$;
- (3) Compute $H_{p_i}(x) = F_{p_i}(x) \times G_{p_i}(x)$ in $\mathbb{F}_{p_i}[x]$ for each $0 \leq i < n$;
- (4) Compute $H(x)$ (which is equal to $F(x) \times G(x)$) by reconstructing its coefficients from $H_{p_0}(x), \dots, H_{p_{n-1}}(x)$ with the CRT algorithm;
- (5) Extract the coefficients of $h(x) = f(x) \times g(x)$ from $H(x)$.

The pseudocode of this algorithm can be found in the original paper of Hart et al. [HTW10, Section 4.1]. Note that their algorithm corresponds to the case $s = 16$. The generalized version of the algorithm for an arbitrary positive integer s can be found in [Mos14a, Section 4.2]. In Theorem 4.1, we give the asymptotic bit-complexity of this algorithm as a function of the polynomial degree (N) and the bundling and bit size parameters.

Theorem 4.1. *Consider two polynomials, $f(x)$ and $g(x)$, both of degree $N - 1$, whose coefficients can be initialized in $O(N)$ bit operations. Using the technique described above, the product $h(x) = f(x) \times g(x)$ can be computed in*

$$(4.16) \quad O\left(Ns(\log(Bs))^{2+\varepsilon} + Ns\left(\log \frac{N}{B}\right)^{1+\varepsilon}\right)$$

bit operations, where B is the bundling parameter, and s is the bit size parameter.

Proof. We analyze each of the five stages of the algorithm. The computation of bundled polynomials $F(x)$ and $G(x)$ in stage (1) consists of sequential applications of logical shifts and ORs, and requires $O(N)$ bit operations. Each bundled polynomial has N/B coefficients, so the multimodular reduction phase (2) requires N/B reductions modulo n primes p_0, \dots, p_{n-1} . We use a *remainder tree* to reduce

each coefficient C of a bundled polynomial modulo p_0, \dots, p_{n-1} . This technique allows us to compute $C \bmod p_0, C \bmod p_1, \dots, C \bmod p_{n-1}$ in $O(t(\log t)^{2+\varepsilon})$ bit operations, where t is the total number of bits in C, p_0, \dots, p_{n-1} [BM74, Section 3]. Since each coefficient of a bundled polynomial fits into Bs bits, we conclude that the multimodular reduction phase requires

$$O\left(\frac{N}{B} \cdot t(\log t)^{2+\varepsilon}\right) = O\left(\frac{N}{B} \cdot Bs(\log(Bs))^{2+\varepsilon}\right) = O\left(Ns(\log(Bs))^{2+\varepsilon}\right).$$

bit operations.

In stage (3), the multiplication of n pairs of polynomials of degree $N/B - 1$ is performed with the Schönhage-Strassen algorithm [GG03, Sections 8.2 – 8.4]. This algorithm requires $O(N \log N \log \log N)$ bit operations to multiply two polynomials of degree N . Hence, stage (3) requires

$$O\left(n \frac{N}{B} \log \frac{N}{B} \log \log \frac{N}{B}\right) = O\left(Bs \frac{N}{B} \log \frac{N}{B} \log \log \frac{N}{B}\right) = O\left(Ns \left(\log \frac{N}{B}\right)^{1+\varepsilon}\right)$$

bit operations.

Finally, consider stages (4) and (5), i.e., the CRT reconstitution and extraction of coefficients. Though the latter involves certain sophisticated techniques, it simply iterates over all N coefficients of the resulting polynomial $H(x)$, and therefore requires $O(N)$ bit operations. Now, consider the CRT reconstitution in stage (4). For the CRT, we use the divide-and-conquer technique [HTW10, Section 4]. For n_1 integer coefficients of size n_2 bits, this approach allows us to complete the restoration of a coefficient in $O(n_2(\log(n_1 n_2))^{2+\varepsilon})$ bit operations. In our case, n_2 is constant and $n_1 = n$, where n is the number of primes in use. In total, there are N/B coefficients to restore, which means that the number of bit operations required is in

$$O\left(\frac{N}{B}(\log n)^{2+\varepsilon}\right) = O\left(\frac{N}{B}(\log(Bs))^{2+\varepsilon}\right).$$

Since $s > \log^{-1} B$, the asymptotic running time of the initialization phase dominates the running time for the CRT reconstitution phase. Combining the costs for the initialization and multiplication phases yields the result (4.16). \square

Note that the class number tabulation formulas (3.8), (3.9) and (3.10) require *two* polynomial multiplications. For example, in order to determine (3.10), we first have to perform the multiplication $\nabla^2(q) = \nabla(q) \times \nabla(q)$, followed by the computation of $\nabla^3(q) = \nabla^2(q) \times \nabla(q)$. In practice, we use a different approach; that is, we initialize $\vartheta_3^2(q)$ (or $\nabla^2(q)$, or $\nabla^2(q^2)$) *directly*, which allows us to evaluate the formula using one polynomial multiplication instead of two.

We describe the initialization mechanism for the example of $\vartheta_3(q)$. A similar approach can be used to initialize $\nabla(q)$ and $\nabla(q^2)$. We compute the first N coefficients of $\vartheta_3(q)$ block by block, using a certain *partition size* S dividing N . The initialization algorithm for the block of S coefficients from M to $M + S - 1$ requires $O(\sqrt{M + S})$ bit operations, as there are precisely $\lfloor \sqrt{M + S} - \sqrt{S} \rfloor$ perfect squares between M and $M + S - 1$; we can easily iterate over all of them within a single loop. Summing over N/S blocks, we obtain that the initialization of N coefficients

of $\vartheta_3(q)$ requires $O(\sqrt{S}) + O(\sqrt{2S}) + \dots + O(\sqrt{N}) = O(N\sqrt{N}/S)$ bit operations. In order to achieve a linear time for initialization, we choose $S = O(\sqrt{N})$.

In turn, the initialization of a block of coefficients of $\vartheta_3^2(q)$ requires two nested loops, which result in $O(S)$ bit operations. Summing $O(S) + O(2S) + \dots + O(N) = O(N^2/S)$, we conclude that S has to grow proportionally to $N/(\log N)^k$ for some non-negative integer k in order for $\vartheta_3^2(q)$ to be initialized in linear or pseudo-linear time. Of course, this is unreasonable. However, for small N , initializing $\vartheta_3^2(q)$ directly works well in practice, even though it is worse asymptotically than using two sequential polynomial multiplications.

We now state the asymptotic complexity of the complete class number tabulation method without including the initialization costs of $\vartheta_3^2(q)$, $\nabla^2(q)$ or $\nabla^2(q^2)$ mentioned above. We obtain Corollary 4.2 by applying the formula for s in (4.14) to Theorem 4.1.

Corollary 4.2. *The class number tabulation algorithm requires*

$$(4.17) \quad O\left(N \log N (\log B)^{2+\varepsilon} + N \log N \left(\log \frac{N}{B}\right)^{1+\varepsilon}\right)$$

bit operations.

In theory, all steps of the algorithm can be parallelized trivially, yielding a speed-up of T using T threads — see [Mos14a, Chapter 4] for a complete description and analysis. In practice, such optimal speedup is difficult to achieve due to the cost of managing the threads and the assumption that all disk I/O is being done in parallel, both reading *and* writing. Special hard disks designed for large-scale parallel applications, such as those used in our experiments, are necessary to get the most out of parallelization.

The method described by Ramachandran et. al. [JRW06, Ram06] has bit complexity $O(|\Delta|^{1/4+\varepsilon})$ for each discriminant, and thus $O(N^{5/4+\varepsilon})$ for all $|\Delta| < N$, including the ERH-verification step. Our new algorithm computes the class numbers asymptotically faster, but we can only use it for $\Delta \not\equiv 1 \pmod{8}$ and require a more expensive method for the remaining congruence class. In addition, Ramachandran's method also computes the class group structures. We describe our approach to this part of the problem in the next section.

5. UNCONDITIONAL CLASS GROUP TABULATION

The class number tabulation technique, described in Sections 3 and 4, allows us to compute unconditionally all class numbers $h(\Delta)$ with $\Delta \not\equiv 1 \pmod{8}$ and $|\Delta| < N$. To resolve the structure of each class group Cl_Δ , we use the algorithm due to Buchmann, Jacobson, and Teske (BJT) [BJT97, Algorithm 4.1], suitable for any generic group G . This algorithm iteratively builds up the set of generators α of G , and terminates whenever the size of the subgroup $\langle \alpha \rangle$ generated by α matches $|G|$.

Note that tabulating class numbers for $\Delta \not\equiv 1 \pmod{8}$ has another major advantage, aside from the fact that we were able to produce the size of each Cl_Δ unconditionally and did not require an additional verification step. Given the factorization of $h(\Delta) = p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_k^{e_k}$, we can ignore those primes p_i for $1 \leq i \leq k$ which have $e_i = 1$, as it means that the p_i -group of Cl_Δ is guaranteed to be cyclic.

We can therefore resolve the structure of a smaller subgroup G of Cl_Δ , satisfying

$$|G| = \prod_{p_i^2 | h(\Delta)} p_i^{e_i}.$$

In practice, $|G|$ was much smaller than $h(\Delta)$ frequently, so this method worked very well.

For $\Delta \equiv 1 \pmod{8}$, where our tabulation method does not produce any class numbers, we used the same method as in [JRW06]. The Buchmann-Jacobson-Teske algorithm can still be used to compute class groups without knowing the class numbers a priori. In this case, it is sufficient to provide a lower bound h^* such that $h^* \leq h(\Delta) \leq 2h^*$ in order to be certain that the whole group was generated — once the size of the subgroup $\langle \alpha \rangle$ generated by α exceeds h^* , we know that we have the entire group.

As described in [JRW06], the main issue with this approach is that the best method to determine the lower bound h^* requires the ERH-dependent averaging method of Bach [Bac90]. To eliminate the ERH dependency, we again followed [JRW06] and applied the Eichler-Selberg trace formula. This formula gives an expression for the trace of the Hecke operator T_n acting on the space of cusp forms $S_k(\Gamma_0(N), \chi)$. When applied to the case where $k = 2$, $N = 1$, and χ the trivial character, the trace formula reduces to the following equality involving Hurwitz class numbers:

$$(5.18) \quad H(4n) + 2 \sum_{t=1}^{\lfloor \sqrt{4n} \rfloor} H(4n - t^2) = 2 \left(\sum_{\substack{d|n \\ d \geq \sqrt{n}}} d \right) - \sigma(n)\sqrt{n} + \frac{1}{6}\chi(n),$$

where $\sigma(n)$ is the indicator function, which is 1 whenever n is a perfect square and 0 otherwise [JRW06, Formula 2.2]. Due to the nature of the BJT algorithm, the size of the class group computed will always divide $h(\Delta)$. Therefore, if one or more of our computed class numbers are wrong, then (5.18) will detect this because the left hand side will be less than the right hand side. Note that in our case the only class numbers $h(\Delta)$ that require verification are those with $\Delta \equiv 1 \pmod{8}$, so it is sufficient in our case to verify that the equality (5.18) holds only for *even* values of n .

One method to use (5.18) to verify all $h(\Delta_1)$ with Δ_1 fundamental and $|\Delta_1| < N$, as suggested in [JRW06], is to first compute the smallest set of n values such that every fundamental discriminant Δ_1 divides at least one Hurwitz class number in the formula. However, a more efficient approach was later suggested by Ramachandran [Ram06], based on simplifying the computation of (5.18) for all values of n between 1 and $\lfloor N/4 \rfloor$.

Following Ramachandran [Ram06, Formulas 4.10, 4.11], but adjusting for the fact that we only need to verify discriminants congruent to 1 mod 8, we define two quantities, LHS and RHS , as follows:

$$(5.19) \quad LHS = \left(\sum_{\substack{\Delta \equiv 0 \pmod{8} \\ |\Delta| \leq 8X}} H(|\Delta|) \right) + 2 \left(\sum_{\substack{\Delta \equiv 0,1 \pmod{4} \\ |\Delta| \leq 8X}} r(\Delta, X) H(|\Delta|) \right);$$

$$(5.20) \quad RHS = \sum_{n=1}^X \left(2 \left(\sum_{\substack{d|2n \\ d \geq \sqrt{2n}}} d \right) - \chi(2n)\sqrt{2n} + \frac{1}{6}\chi(2n) \right).$$

Here, $r(\Delta, X)$ counts the number of solutions to the equation $\Delta = t^2 - 8n$ for $1 \leq n \leq X$:

$$(5.21) \quad r(\Delta, X) = \begin{cases} 0, & \text{if } \Delta \equiv 5 \pmod{8}; \\ \lfloor (Y+1)/2 \rfloor, & \text{if } \Delta \equiv 1 \pmod{8}; \\ \lfloor (Y+2)/4 \rfloor, & \text{if } \Delta \equiv 4 \pmod{8}; \\ \lfloor Y/4 \rfloor, & \text{if } \Delta \equiv 0 \pmod{8}, \end{cases}$$

where $Y = \lfloor \sqrt{8X + \Delta} \rfloor$. We computed both LHS and RHS in parallel for $X = \lfloor N/8 \rfloor$, where $N = 2^{40}$. The expression LHS is evaluated using the table of class numbers of fundamental discriminants computed using the BJT method; see [Ram06, Algorithm 4.1] for pseudocode. Though computationally more intensive, the calculation of the RHS is more straightforward and easily parallelizable. In order to compute the divisors for each $n \leq X$, we use the formula (5.20) in conjunction with a segmented sieve.

6. PERFORMANCE

For the class number tabulation using out-of-core polynomial multiplication, we used the FLINT library for number theory, maintained by Hart [Har14]. In particular, we used the `nmod_poly_mullow` routine for polynomial multiplication in $\mathbb{F}_p[x]$. The FLINT library also contains subroutines for fast reduction modulo primes p_0, \dots, p_{n-1} and CRT reconstitution, respectively. We used OpenMP for parallelization.

For the class group computation, we used Sayles's libraries `optarith` and `qform`, which contain fast implementations of binary quadratic form arithmetic [Say13a, Say13b], including implementations targeted to machine-size operands that avoid multi-precision integer arithmetic. We also use Message Passing Interface (MPI) for parallelization. The source code for our program can be found in [Mos14b].

Our computations were performed on WestGrid's supercomputer Hungabee, located at the University of Alberta, Canada [Wes14]. Hungabee is a 16 TB shared memory system with 2048 Intel Xeon cores, 2.67GHz each. Each user of Hungabee may request at most 8 GB of memory per core. Also, Hungabee provides a high performance 53 TB storage space, which allows to write to multiple disks in parallel. Note that the fast disk I/O requirement is essential for the high performance of our program.

We first discuss our class number tabulation program. We performed three polynomial multiplications, described in formulas (3.8), (3.9) and (3.10). After running several tests, we determined that Hungabee can comfortably multiply polynomials of 2^{25} coefficients without requiring additional memory. This observation allowed us to make the proper choice of a bundling parameter B .

Table 1 contains the list of parameters which we used for our computations, and the amount of disk space needed to store intermediate computations required for the polynomial multiplication. Here, C is the bound on $H(|\Delta|)$ defined in (4.13), s is the bit size parameter (4.14), and n is the number of 63-bit primes required

TABLE 1. Computational parameters

Formula	Δ	N	B	C	s	n	Disk space
$\nabla^2(q^2) \cdot \vartheta_3(q)$	8 (mod 16)	2^{36}	2^{11}	11199314	24	1586	859 GB
$\vartheta_3^2(q) \cdot \nabla(q^2)$	12 (mod 16)	2^{36}	2^{11}	11199314	25	1652	893.4 GB
$\nabla^2(q) \cdot \nabla(q)$	5 (mod 8)	2^{37}	2^{12}	21381515	26	3435	1855 GB

for correct CRT reconstitution (4.15). For each multiplication, we requested 64 processors and 8 GB of memory per core. The number of files was chosen to be $m = 4096$. Table 2 lists timings for each of the three class number tabulation algorithms.

TABLE 2. Timings for the class number tabulation program

$f(x)$	$g(x)$	CPU time	Real time
$\nabla^2(q^2)$	$\vartheta_3(q)$	23d 11h 10m 56s	8h 47m 59s
$\vartheta_3^2(q)$	$\nabla(q^2)$	29d 21h 2m 56s	11h 12m 14s
$\nabla^2(q)$	$\nabla(q)$	68d 5h 8m 16s	25h 34m 49s

Table 3 contains timings for computing the class group structures. As expected, for $\Delta \equiv 1 \pmod{8}$ our program takes significantly more time, since Ramachandran's approach requires the computation of the whole group. If we assume that all Δ were handled using solely Ramachandran's technique, then 64 processors would complete the (conditional) tabulation to 2^{40} in 80d 11h 9m 48s, as opposed to 31d 22h 45m 8s (counting the class number tabulation and the verification). Note that 81.13% of time in our computations was spent on the computation of Cl_Δ for $\Delta \equiv 1 \pmod{8}$ and the verification of the result.

TABLE 3. Timings for the class group tabulation program

Δ	CPU time	Real time	# processors
$\Delta \not\equiv 1 \pmod{8}$	267d 4h 31m 40s	4d 3h 26m 44s	64
$\Delta \equiv 1 \pmod{8}$	1657d 22h 12m 6s	39h 28m 27s	1008

We also observe that the structures of Cl_Δ for all congruence classes with $\Delta \not\equiv 1 \pmod{8}$ were computed over 6.25 times faster than those with $\Delta \equiv 1 \pmod{8}$. If we include the verification cost for the 1 mod 8 case and the class number tabulation for the rest, then the entire computation for all $\Delta \not\equiv 1 \pmod{8}$ is roughly 4.72 times faster than that for 1 mod 8. Such a significant speedup occurs due to the fact that in 57.34% of the cases $h(\Delta)$ had a square-free factor exceeding $\sqrt{h(\Delta)}$, which means that the size of the subgroup that we had to resolve was small relative to the size of the group itself. Moreover, in 1.67% of the cases $h(\Delta)$ were square-free, which means that no resolution of class groups was needed at all. In general, over 85.13% of class numbers $h(\Delta)$ possessed a square-free part larger than 1. In Table 4, we present the counts of class numbers up to 2^{40} with various divisibility properties. In particular, column 3 counts class numbers with square-free part greater than 1, column 4 counts $h(\Delta)$ with square-free part exceeding $\sqrt{h(\Delta)}$, and

TABLE 4. Counts of $h(\Delta)$ with various divisibility properties

Δ	Total Δ	$p \mid h, p^2 \nmid h$	$h = g^2e, e > \sqrt{h}$	square-free h
8 (mod 16)	55701909754	47077629143	32012088117	941347842
12 (mod 16)	55701909855	47091713960	31927265003	915383075
5 (mod 8)	111403819688	95517292502	63828635213	8828052571
1 (mod 8)	111403819373	94502061670	55851403024	7295483368

column 5 counts class numbers that are square-free. Our data is separated into four congruence classes.

Note that the counts for $\Delta \equiv 1 \pmod{8}$ were not included in the percentages listed above. The counts are similar to the other congruence classes, but divisibility properties of the class number played no role in the computation for the 1 mod 8 case as the class numbers were not computed first. It should be emphasized that the rapid computation of all class numbers using theta-series is what allowed us to take advantage of these properties when resolving the group structures.

Finally, we compare the performance of our program to the implementation of Ramachandran [Ram06], and the `quadclassunit0` routine of the PARI/GP library [Par14]. For the class group resolution, the latter implementation uses Hafner and McCurley’s subexponential index calculus algorithm [McC89, HMC89]. For each implementation, we used a single Intel Xeon 2.27GHz processor to compute Cl_Δ for every fundamental $\Delta < 0$ such that $|\Delta|$ lies in the interval from 2^{39} to $2^{39} + 2^{20}$. For this computation, we used the ERH-dependent version of the BJT algorithm for our implementation and that of Ramachandran (i.e., no prior class number tabulation nor verification in either case). The resulting timings are listed in Table 5.

TABLE 5. Timings for various class group tabulation implementations

$ \Delta_{min} $	$ \Delta_{max} $	Total Δ	Our program	[Ram06]	PARI/GP
2^{39}	$2^{39} + 2^{20}$	318729	438s	736s	1181s

All three implementations yield correct results under the assumption of the ERH. Though our program and Ramachandran’s implementation use the same algorithm, it significantly outperforms the latter. We believe that the optimized binary quadratic form arithmetic in Sayles’s libraries `optarith` and `qform` [Say13a, Say13b] used in our program accounts for the improvement.

Note that asymptotically the Hafner-McCurley algorithm (with subexponential complexity in $\log |\Delta|$) is superior to the BJT algorithm (exponential complexity). Thus, although it should be faster for a sufficiently large discriminant bound, our results show that the bound 2^{40} is still below the crossover point.

7. NUMERICAL RESULTS

7.1. Bounds on $L(1, \chi_\Delta)$. In 1928, Littlewood [Lit28] demonstrated that, assuming the ERH,

$$(7.22) \quad (1 + o(1))(c_1 \log \log |\Delta|)^{-1} < L(1, \chi_\Delta) < (1 + o(1))c_2 \log \log |\Delta|,$$

where

$$c_1 = \frac{12e^\gamma}{\pi^2} \text{ and } c_2 = 2e^\gamma \text{ when } 2 \nmid \Delta;$$

$$c_1 = \frac{8e^\gamma}{\pi^2} \text{ and } c_2 = e^\gamma \text{ when } 2 \mid \Delta,$$

and $\gamma \approx 0.57722$ is the Euler-Mascheroni constant. Later, Shanks studied these bounds more carefully by defining two quantities,

$$ULI = \frac{L(1, \chi_\Delta)}{c_2 \log \log |\Delta|} \text{ and } LLI = L(1, \chi_\Delta) c_1 \log \log |\Delta|,$$

and ignoring $o(1)$ term in Littlewood's estimates [Sha73]. These quantities allow us to test whether Littlewood's bounds are violated, for if the ERH does not hold, then for large $|\Delta|$ we might find $ULI > 1$ or $LLI < 1$. Note that there *are* small Δ such that $ULI > 1$ or $LLI < 1$, namely $\Delta = -3, -4, -163$. We assume that values of ULI and LLI for these discriminants are largely influenced by $o(1)$ terms. Aside from $\Delta = -3, -4, -163$, we did not find any occurrences of discriminants which violate Littlewood's bounds. Ignoring these Δ , the largest $ULI \approx 0.85183$ corresponds to $\Delta = -8$, and the smallest $LLI \approx 1.00944$ corresponds to $\Delta = -232$.

In addition to Littlewood's bounds, we also studied the growth of $L(1, \chi_\Delta)$. In Table 6, we list successive maximas of $L(1, \chi_\Delta)$ which did not occur in Table 5.3 of [Ram06]. The last discriminant found was $\Delta = -685122125399$, which corresponds to the largest $L(1, \chi_\Delta) \approx 8.47178$ with $|\Delta| < 2^{40}$. As for successive minimas of $L(1, \chi_\Delta)$, no new discoveries were made. The smallest $L(1, \chi_\Delta) \approx 0.17070$ corresponds to $\Delta = -107415709003$.

TABLE 6. Successive $L(1, \chi_\Delta)$ maxima

$ \Delta $	$L(1, \chi_\Delta)$	ULI
210015218111	8.26604	0.71164
332323080311	8.30989	0.71161
503494619759	8.31253	0.70848
603231310919	8.32466	0.70807
685122125399	8.47178	0.71957

7.2. The Cohen-Lenstra Heuristics. In 1984, Cohen and Lenstra presented several powerful heuristics on the structure of the odd part of the class group Cl_Δ [CL84]. The odd part Cl_Δ^* is simply the largest subgroup of Cl_Δ with an *odd* cardinality.

Conjecture 7.1 ([CL84, C1]). *Define*

$$\eta_k(l) = \prod_{i=1}^k \left(1 - \frac{1}{l^i}\right) \text{ and } C_\infty = \prod_{i=2}^{\infty} \zeta(i) \approx 2.294856589,$$

where $\zeta(s)$ denotes the Riemann zeta function. For $\Delta < 0$, the probability that the odd part of the class group Cl_Δ is cyclic is

$$(7.23) \quad \Pr(Cl_\Delta^* \text{ is cyclic}) = \frac{315\zeta(3)}{6\pi^4\eta_\infty(2)C_\infty} \approx 0.977575.$$

Conjecture 7.2 ([CL84, C2]). *Let l be an odd prime. For $\Delta < 0$, the probability that l divides $h(\Delta)$ is*

$$(7.24) \quad \Pr(l \mid h(\Delta)) = 1 - \eta_\infty(l).$$

Conjecture 7.3 ([CL84, C5]). *Let l be an odd prime. For $\Delta < 0$, the probability that the l -rank of Cl_Δ is equal to r is*

$$(7.25) \quad \Pr(l\text{-rank} = r) = \frac{\eta_\infty(l)}{l^{r^2} \eta_r(l)^2}.$$

In order to study these conjectures, we follow the approach of Jacobson et al. and introduce three functions: $c(x)$, $p_l(x)$ and $p_{l,r}(x)$ [JRW06, Section 3.2]:

$$c(x) = \frac{\# \text{ of } Cl_\Delta^* \text{ cyclic with } |\Delta| < x}{\# \text{ of } \Delta \text{ with } |\Delta| < x} / \Pr(Cl_\Delta^* \text{ is cyclic});$$

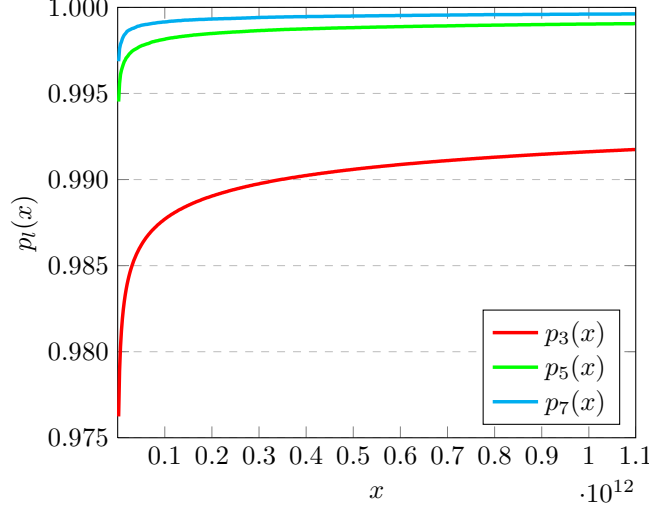
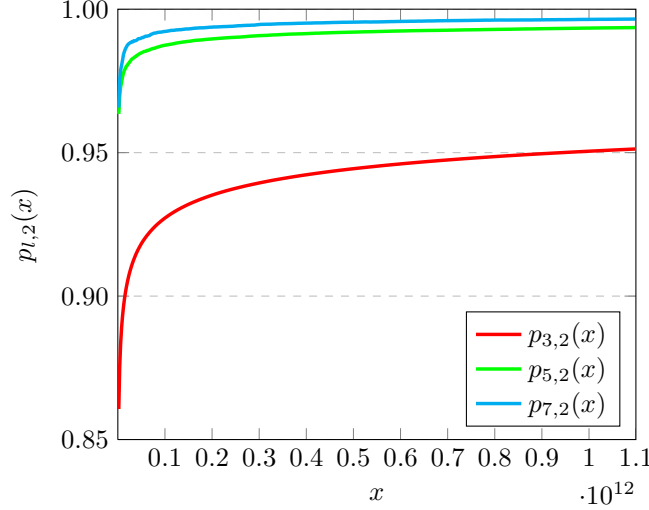
$$p_l(x) = \frac{\# \text{ of } h(\Delta) \text{ divisible by } l \text{ with } |\Delta| < x}{\# \text{ of } \Delta \text{ with } |\Delta| < x} / \Pr(l \mid h(\Delta));$$

$$p_{l,r}(x) = \frac{\# \text{ of } Cl_\Delta \text{ with } l\text{-rank} = r \text{ and } |\Delta| < x}{\# \text{ of } \Delta \text{ with } |\Delta| < x} / \Pr(l\text{-rank} = r).$$

If the Cohen-Lenstra heuristics hold, we would expect each of those functions to approach 1 as x grows. We observe this behavior in Figures 1 and 2, which plot $p_l(x)$ and $p_{l,2}(x)$ for $l = 3, 5, 7$, respectively. The values of $c(x)$, as well as the counts of non-cyclic Cl_Δ^* , are presented in Table 7. Note that our counts differ from the ones given in [JRW06, Table 3]. For example, the total number of non-cyclic Cl_Δ^* for $|\Delta| < 10^{11}$ given in [JRW06, Table 3] is 603101904, whereas our count in Table 7 suggests that this number is 636501087. In general, our counts are over 1.044 times larger than the counts given in [JRW06, Table 3]; this ratio grows with x and reaches 1.055 for $x = 10^{11}$. We argue that values in Table 7 are correct, because the output of algorithms for small $N < 10^9$ matches that of PARI/GP [Par14]. Finally, in Tables 8 and 9 we count the total number of $h(\Delta)$ divisible by a prime l , and class groups with a certain l -rank.

TABLE 7. Number of noncyclic odd parts of class groups

x	Total	Non-cyclic	Percent	$c(x)$
10^{11}	30396355052	636501087	2.09400	1.00152
$2 \cdot 10^{11}$	60792710179	1283029629	2.11050	1.00135
$3 \cdot 10^{11}$	91189065248	1932535723	2.11926	1.00126
$4 \cdot 10^{11}$	121585420327	2583844783	2.12513	1.00120
$5 \cdot 10^{11}$	151981775550	3236429002	2.12948	1.00116
$6 \cdot 10^{11}$	182378130683	3889995513	2.13293	1.00112
$7 \cdot 10^{11}$	212774486110	4544337515	2.13575	1.00109
$8 \cdot 10^{11}$	243170840635	5199342505	2.13814	1.00107
$9 \cdot 10^{11}$	273567195607	5854902775	2.14021	1.00105
10^{12}	303963550712	6510933430	2.14201	1.00103
2^{40}	334211458670	7164219493	2.14362	1.00101

FIGURE 1. Values of $p_l(x)$ FIGURE 2. Values of $p_{l,2}(x)$ 

7.3. First Occurrences of Non-cyclic p -Sylow Subgroups. During our computations, we also looked at the problem of finding Cl_Δ with the smallest $|\Delta|$ which corresponds to a certain p -group structure. This question was explored by Buell in [Bue99], where he tabulated the first occurrences of what he called “exotic” groups. He gave a list of first even and odd Δ , as well as the total number of them up to $2.2 \cdot 10^9$. This list was extended by Ramachandran to $2 \cdot 10^{11}$ [Ram06]. In Tables 10, 11 and 12 we further extend Ramachandran’s results by listing first occurrences of class groups that are not present in Tables 5.13, 5.15 and 5.17 of [Ram06]. Previously unknown minimal discriminants whose class groups have a variety of exotic

TABLE 8. Counts of class numbers divisible by l

x	$3 \mid h$	$5 \mid h$	$7 \mid h$	$11 \mid h$	$13 \mid h$
10^{11}	13206088529	7271547905	4956628127	3011896994	2516050182
$2 \cdot 10^{11}$	26447989308	14547903930	9914941601	6025009729	5032948550
$3 \cdot 10^{11}$	39700741936	21825546084	14873726078	9038458883	7550137579
$4 \cdot 10^{11}$	52959934649	29103662856	19832681021	12052003780	10067454468
$5 \cdot 10^{11}$	66223739128	36382211005	24791661364	15065606774	12584840703
$6 \cdot 10^{11}$	79491008890	43661126382	29750874514	18079320114	15102137499
$7 \cdot 10^{11}$	92761033879	50940277442	34710302571	21092999797	17619561852
$8 \cdot 10^{11}$	106033521908	58219944093	39669843978	24106720004	20137035912
$9 \cdot 10^{11}$	119308020675	65499671827	44629193028	27120432707	22654498276
10^{12}	132584350621	72779583545	49588756987	30134192653	25171929972
2^{40}	145797270882	80023955398	54524158518	33133247297	27677104824

TABLE 9. Counts of class groups with l -rank = r

x	$r = 2$			$r = 3$		
	$l = 3$	$l = 5$	$l = 7$	$l = 3$	$l = 5$	$l = 7$
10^{11}	554992183	61905528	14909598	1891327	19701	824
$2 \cdot 10^{11}$	1119549000	124086783	29864434	3941440	39455	1699
$3 \cdot 10^{11}$	1686937952	186346310	44837690	6041677	59455	2555
$4 \cdot 10^{11}$	2256067209	248638170	59813385	8170672	79056	3392
$5 \cdot 10^{11}$	2826419025	310963856	74791724	10319592	99020	4302
$6 \cdot 10^{11}$	3397716149	373303706	89772515	12486498	119058	5158
$7 \cdot 10^{11}$	3969781768	435637308	104762170	14667860	138969	6050
$8 \cdot 10^{11}$	4542454057	498010970	119754407	16861780	158992	6949
$9 \cdot 10^{11}$	5115675246	560398913	134735076	19064061	179000	7804
10^{12}	5689326792	622806579	149727575	21274374	199005	8677
2^{40}	6260628955	684906543	164647966	23481723	218977	9586

structures were discovered, including $\Delta = -824746962451$ which is the smallest discriminant in absolute value with 17-rank equal to three.

We also looked at the first occurrences of doubly and trebly non-cyclic class groups. One of the most interesting discoveries is $\Delta = -658234953151$ with $Cl_{\Delta}^* \cong C(5 \cdot 7 \cdot 17) \times C(5 \cdot 7 \cdot 17)$, where $C(x)$ denotes the cyclic group of order x . In Tables 13 and 14, we list first occurrences of doubly and trebly non-cyclic p -groups that are not present in Tables 5.18 and 5.19 of [Ram06].

The complete tables with all frequency counts for discriminants satisfying $|\Delta| < 2^{40}$ can be found in [Mos14a]. The data is soon to appear online on The L -functions and Modular Forms Database [LMFDB].

TABLE 10. Non-cyclic rank 2 p -Sylow subgroups

p	e_1	e_2	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	7	5	*	*	253237383431	2
3	8	4	*	*	225796561799	10

Table 10 – continued from previous page

p	e_1	e_2	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	10	2	1018482429656	2	65798421911	908
3	10	3	*	*	766483839959	2
3	11	1	786365476244	16	52623967679	7879
3	11	2	*	*	677250946319	24
3	12	1	*	*	512068796879	177
5	5	3	*	*	213265691687	15
5	6	2	775319038196	5	75913193999	175
5	7	1	573881434136	107	48662190359	4626
5	8	1	*	*	941197327199	3
7	3	3	798957687128	2	40111506371	10
7	5	2	*	*	336699684383	5
11	3	2	344379903284	5	91355041631	29
11	5	1	*	*	935094698711	2
13	3	2	*	*	366445322799	2
13	4	1	604812537944	15	55385334839	522
17	2	2	522715590248	3	94733724779	12
17	4	1	*	*	607531396391	7
23	3	1	428918887976	12	74447537447	296
29	3	1	*	*	323459074199	19
31	3	1	*	*	503905534439	14
53	2	1	313806056276	24	34862413351	200
59	2	1	278155567784	6	65887828631	81
61	2	1	388888967156	6	148712371111	62
67	2	1	323124297044	3	131240605511	28
73	2	1	*	*	350771311831	17
83	2	1	*	*	589364144599	3
89	2	1	*	*	619130566127	2
97	2	1	*	*	438994809599	2
101	2	1	*	*	981198752759	1
223	1	1	229260698804	17	36799898071	49
227	1	1	248738329160	17	129251563279	43
241	1	1	275897077784	13	74882513855	33
251	1	1	274131019432	7	78181110431	24
263	1	1	482147329592	7	37893813311	31
269	1	1	241103392196	4	11129396567	22
271	1	1	291445797352	5	171753801031	18
277	1	1	266610558308	6	128621435167	18
281	1	1	644634989492	2	266379885935	13
293	1	1	874615243688	3	158602460567	17
307	1	1	749662659128	3	149654057447	13
311	1	1	666221368184	4	111301462879	8
313	1	1	416363928728	3	303265490831	14
331	1	1	158739065384	3	388995885319	7
337	1	1	506841655124	2	283026340679	6

Table 10 – continued from previous page

p	e_1	e_2	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
349	1	1	804641768168	1	32819826815	5
353	1	1	839537284648	2	305328598259	9
359	1	1	*	*	627072510479	4
373	1	1	*	*	215425181891	5
379	1	1	*	*	356510006687	4
383	1	1	878382375224	1	137740312007	6
397	1	1	*	*	434530437127	2
409	1	1	*	*	594857692087	1
421	1	1	*	*	422660888879	4
431	1	1	*	*	567134500223	3
433	1	1	*	*	791181108079	2
439	1	1	*	*	782761871063	2
443	1	1	462953812184	2	146805555551	2
449	1	1	*	*	347760731679	3
457	1	1	212262246356	1	413877350951	2
461	1	1	*	*	353455619411	4
463	1	1	1047876328724	1	679010903567	1
467	1	1	683325795752	1	817093587359	1
503	1	1	*	*	544027580079	1
521	1	1	969683875304	1	363850136623	1
577	1	1	*	*	733117084823	1
719	1	1	*	*	737463696271	1

TABLE 11. Non-cyclic rank 3 p -Sylow subgroups

p	e_1	e_2	e_3	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	3	3	2	341946799364	2	20687610651	11
3	4	3	2	295863285976	3	744853350587	1
3	5	2	2	412703787940	9	45248632247	24
3	5	4	1	186447381556	3	376544421947	5
3	6	2	2	582608055992	1	9483757583	9
3	6	4	1	900453600692	1	276331426207	2
3	7	2	2	*	*	484468933679	1
3	7	3	1	1076743681124	1	338926563823	10
3	8	2	1	276573602516	12	59714529551	139
3	9	1	1	182514096404	127	12792023879	978
3	9	2	1	*	*	581116399159	14
3	10	1	1	989021051864	1	146114436719	104
3	11	1	1	*	*	797107037711	1
5	4	2	1	204195796664	3	116279191211	7
5	6	1	1	*	*	349008665407	5
7	2	2	1	439240920004	1	868770849819	3
7	4	1	1	356820088964	1	451900165735	4
11	2	1	1	889484965924	2	145931588651	9

Table 11 – continued from previous page

p	e_1	e_2	e_3	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
13	1	1	1	218639119912	11	38630907167	20
17	1	1	1	*	*	824746962451	2

TABLE 12. Non-cyclic rank 4 p -Sylow subgroups

p	e_1	e_2	e_3	e_4	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	3	3	1	1	*	*	1074734433547	1
3	4	2	1	1	426126877012	3	128589208863	8
3	5	2	1	1	*	*	473827747963	2
3	6	1	1	1	460093393912	8	76951070303	15
3	7	1	1	1	1047320556596	1	513092626699	2
3	8	1	1	1	*	*	226138531999	2

TABLE 13. Doubly non-cyclic class groups

p_1	p_2	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	83	411040250696	8	50476998239	69
3	89	271776528392	14	146604777199	29
3	97	373716927704	5	43344787079	22
3	101	204919229864	3	270845549231	12
3	103	374301791476	8	93069031703	14
3	107	747657517988	2	193384461719	15
3	109	379370724596	5	35029686023	17
3	127	761263140536	1	127466536019	10
3	131	*	*	248486020319	2
3	137	*	*	373309196719	4
3	139	*	*	261265037799	3
3	149	*	*	555574557467	4
3	157	*	*	258504106919	2
3	163	*	*	288700332223	5
3	191	*	*	778133573263	1
3	193	*	*	4 15837893871	1
3	197	*	*	675588676571	1
3	223	*	*	1044678632711	1
5	47	337410526616	16	8182208159	78
5	53	375201391636	8	22759605719	28
5	59	842452697976	2	166413410411	20
5	61	621148062232	2	198540663599	14
5	67	952877473160	1	202658297511	13
5	79	*	*	695299489415	4
5	83	*	*	255558978287	5
5	97	*	*	957408127639	1
5	107	*	*	895542638663	1

Table 13 – continued from previous page

p_1	p_2	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
7	37	220308406520	5	49918973471	36
7	43	395768104936	1	57006644887	18
7	47	611628524996	2	98533572251	12
7	53	819974042456	1	532593252151	6
7	59	*	*	746029216663	3
7	61	*	*	530458082031	2
7	79	*	*	1010896284767	1
7	101	*	*	613532171711	1
11	19	293745669956	33	19439678123	86
11	23	440245788692	7	94266055451	45
11	29	258828614756	2	246806029679	13
11	31	752299766228	1	167546860535	6
11	37	*	*	507297592171	1
13	23	886308340568	1	303087341987	15
13	31	1042065325544	1	309693265351	5
13	37	*	*	583833769207	1
13	41	969016080404	1	407911409771	2
17	19	150334566104	2	473841789911	9
17	23	432363302164	2	54134972891	3
17	29	*	*	892052200651	2
17	31	*	*	1035367542059	1
19	23	*	*	659380117199	1
19	29	*	*	915336787039	1

TABLE 14. Trebly non-cyclic class groups

p_1	p_2	p_3	First even $ \Delta $	# even Δ	First odd $ \Delta $	# odd Δ
3	5	17	278849168408	19	60235736039	63
3	5	23	703386940456	3	148439200263	14
3	5	29	*	*	300193517399	5
3	5	31	*	*	323714678543	5
3	5	37	*	*	999098015071	1
3	7	23	*	*	805192394183	1
5	7	11	*	*	656450533751	6
5	7	13	786460186856	1	110671542299	3
5	7	17	*	*	658234953151	1

7.4. Euler’s Conjecture on Idoneal Numbers. Consider a discriminant Δ such that Cl_Δ is isomorphic to $(\mathbb{Z}/2\mathbb{Z})^l$ for some $l > 0$. All such Δ are related to so-called *idoneal* numbers, which were studied by Euler and Gauß (see the extensive survey on idoneal numbers by Kani [Kan11]). A positive number D is idoneal if every integer n , which is uniquely representable in the form $n = x^2 \pm Dy^2$ with $\gcd(x^2, Dy^2) = 1$, is either a prime, or a prime power, or twice one of these. Both Euler and Gauß tabulated idoneal numbers, and conjectured that the largest of

them does not exceed 1848 [Gau86, §303]. From the class group perspective, it means that $\Delta = -5460$ is the largest fundamental discriminant such that $Cl_\Delta \cong (\mathbb{Z}/2\mathbb{Z})^l$. In 1918, the hypothesis of Euler and Gauß was confirmed by Hecke and Landau under the assumption of the ERH [Lan18]. However, unconditionally this problem still remains open, though Weinberger was able to prove that there exists *at most* one idoneal number exceeding 1848 [Wei73]. In our computations, we confirm that up to 2^{40} the largest in its absolute value fundamental discriminant Δ with $Cl_\Delta \cong (\mathbb{Z}/2\mathbb{Z})^l$ is $\Delta = -5460$. This result agrees with findings of Euler and Gauß. Note that there exists one non-fundamental discriminant, namely $\Delta = -7392$, which is larger than -5460 in its absolute value and has the group structure as above.

8. FURTHER WORK

Our novel approach to class group tabulation has enabled us to extend the feasibility limit. Pushing our methods further would probably require a class number tabulation mechanism for $\Delta \equiv 1 \pmod{8}$. Presently, no efficient class number tabulation formulas are known for this congruence class. One formula that might be of interest for future exploration is due to Humbert [Wat35, Section 6], who discovered that

$$(8.26) \quad \sum_{n=0}^{\infty} F(8n+7)q^n = S^{-1}(q) \sum_{n=1}^{\infty} (-1)^{n+1} n^2 \frac{q^{\frac{n(n+1)}{2}} - 1}{1 + q^n},$$

where

$$S(q) = \sum_{n=0}^{\infty} (-1)^n (2n+1) q^{\frac{n(n+1)}{2}} = 1 - 3q + 5q^3 - 7q^6 + 9q^{10} - \dots$$

Despite its look, the large series on the right hand side of (8.26) does not take long to initialize, as it is simple to derive the formula for its n -th coefficient. The main problem lies in the computation of $S^{-1}(q)$, which can take a significant amount of time, especially if $S(q)$ is of a high degree. Also, the coefficients of $S(q)$ grow very fast. For example, its 16-th coefficient has bit size 10, 64-th — bit size 59, and 65536-th fits into 1135 bits. These coefficients have to be somehow truncated, for example, by reducing them modulo some prime p , such that $F(8n+7) < p$ for any $n < (N-7)/8$. However, this approach also brings certain difficulties, as it significantly increases the bit size parameter s . Despite all the obstacles, the usage of the formula (8.26) might still be faster than the conditional computation of $\Delta \equiv 1 \pmod{8}$ followed by the verification procedure. We tried to use this approach, and in fact our implementation includes a subroutine `invert` for out-of-core polynomial inversion [Mos14b]. This subroutine utilizes a Newton iteration algorithm [GG03, Algorithm 9.3], which performs the inversion of an arbitrary polynomial to degree $2^n - 1$ by sequentially computing its inverse to degrees $3, 7, \dots, 2^k - 1, \dots, 2^n - 1$ for $2 \leq k \leq n$. Each iteration in this algorithm requires one squaring of a polynomial and one polynomial multiplication. Unfortunately, we were unable to produce class numbers using this method due to the number of difficulties previously mentioned.

We also believe that the class group computation can get accelerated by using Sutherland's p -group discrete logarithm algorithms [Sut11]. The idea is simple: when the class number $h(\Delta)$ is known, instead of computing all of the potentially

non-cyclic subgroups of Cl_Δ we compute the structure of each potentially non-cyclic p -group separately. Sutherland's algorithms may be especially useful when resolving the structure of a 2-group, as we can precompute its rank by factoring Δ . In some cases, the 2-rank allows us to terminate the 2-group resolution earlier by ignoring some of its generators of order 2.

Finally, we note that the question of unconditional tabulation of class groups with positive Δ is still left open, and is currently work in progress.

REFERENCES

- Bac90. E. Bach, *Explicit bounds for primality testing and related problems*, Mathematics of Computation 55, 1990, pp. 355 – 380.
- Bac95. E. Bach, *Improved approximations for Euler products*, Number Theory: CMS Proceedings 15, American Mathematical Society Providence, RI, 1995, pp. 13 – 28.
- Bel24. E. T. Bell, *The class number relations implicit in the Disquisitiones arithmeticae*, Bulletin of the American Mathematical Society 30 (5-6), 1924, pp. 236 – 238.
- BJT97. J. Buchmann, M. J. Jacobson, Jr., E. Teske, *On some computational problems in finite abelian groups*, Mathematics of Computation 66 (220), 1997, pp. 1663 – 1687.
- BM74. A. Borodin, R. T. Moenck, *Fast modular transforms*, Journal of Computer and System Sciences 8, 1974, pp. 366 – 386.
- Bue99. D. A. Buell, *The last exhaustive computation of class groups of complex quadratic number fields*, Number theory, CRM Proceedings and Lecture Notes 19, American Mathematical Society, Providence, RI, 1999, pp. 35 – 53.
- GG03. J. von zur Gathen, J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2nd edition, 2003.
- CL84. H. Cohen, H. W. Lenstra, Jr., *Heuristics on class groups*, Number Theory, Noordwijkerhout, Lecture Notes in Mathematics 1052, Springer-Verlag, New York, 1984, pp. 26 – 36.
- Coh93. H. Cohen, *A course in computational algebraic number theory*, Springer-Verlag, Berlin, 1993.
- Gau86. C. F. Gauß, *Disquisitiones arithmeticae*, 1798. Translated into English by A. A. Clarke and reprinted, Springer-Verlag, 1986.
- Gro85. E. Grosswald, *Representation of Integers as Sums of Squares*, Springer-Verlag, New York, 1985.
- Har14. W. B. Hart, *FLINT: Fast Library for Number Theory*, version 2.4.1, <http://www.flintlib.org>, 2014.
- HMcC89. J. Hafner, K. McCurley, *A rigorous subexponential algorithm for computation of class groups*, Journal of American Mathematical Society 2, 1989, pp. 837 – 850.
- HTW10. W. B. Hart, G. Tornar a, M. Watkins, *Congruent number theta coefficients to 10^{12}* , Algorithmic Number Theory - ANTS-IX (Nancy, France), Lecture Notes in Computer Science 6197, Springer-Verlag, Berlin, 2010, pp. 186 – 200.
- JRW06. M. J. Jacobson, Jr., S. Ramachandran, H. C. Williams, *Numerical results on class groups of imaginary quadratic fields*, Algorithmic Number Theory - ANTS-VII (Berlin, Germany), Lecture Notes in Computer Science 4076, Springer-Verlag, Berlin, 2006, pp. 87 – 101.
- Kan11. E. Kani, *Idoneal numbers and some generalizations*, Annales des Sciences Math matiques du Qu bec 35, 2011, pp. 197 – 227.
- Kro60. L. Kronecker, * ber die Anzahl der verschiedenen classen quadratischer Formen von negativer Determinante*, Journal f r die reine und angewandte Mathematik 57 (4), 1860, pp. 248 – 255.
- Lan18. E. Landau, * ber die Klassenzahl imagin r-quadratischer Zahlk rper*, Ges. Wiss. G ttingen, Math.-Phys., 1918, pp. 285 – 295.
- Lit28. J. E. Littlewood, *On the class number of the corpus $P(\sqrt{-k})$* , Proceedings of the London Mathematical Society 27, 1928, pp. 358 – 372.
- LMFDB. The LMFDB Collaboration, *The L-functions and Modular Forms Database*, <http://www.lmfdb.org>, 2015.
- McC89. K. McCurley, *Cryptographic key distribution and computation in class groups*, Proceedings of NATO ASI Number Theory and Applications, Kluwer Academic Publishers, 1989, pp. 459 – 479.

- Mey12. D. Meyer, *The second p -class group of a number field*, International Journal of Number Theory 2 (8), 2012, pp. 471 – 505.
- Mos14a. A. S. Mosunov, *Unconditional Class Group Tabulation to 2^{40}* , Master's thesis, University of Calgary, Calgary, Alberta, 2014.
- Mos14b. A. S. Mosunov, *Class group tabulation program*, <https://github.com/amosunov>, 2014.
- Par14. The PARI Group, *PARI/GP version 2.7.1*, <http://pari.math.u-bordeaux.fr>, Bordeaux, 2014.
- Ram01. O. Ramaré, *Approximate formulae for $L(1, \chi)$* , Acta Arithmetica 100, 2001, pp. 245 – 266.
- Ram06. S. Ramachandran, *Class groups of quadratic fields*, Master's thesis, University of Calgary, Calgary, Alberta, 2006.
- Say13a. M. Sayles, *Improved arithmetic in the ideal class group of imaginary quadratic fields with an application to integer factoring*, Master's thesis, University of Calgary, Calgary, Alberta, 2013.
- Say13b. M. Sayles, C libraries `optarith` and `qform` for fast binary quadratic form arithmetic, <https://github.com/maxwellsayles>, 2013.
- Sha73. *Systematic examination of Littlewood's bounds on $L(1, \chi)$* , Proceedings of Symposia in Pure Mathematics, AMS, Providence, R.I., 1973, pp. 267 – 283.
- Sut11. A. V. Sutherland, *Structure computation and discrete logarithms in finite abelian p -groups*, Mathematics of Computation 80 (273), 2011, pp. 477 – 500.
- SvdV91. R. Schoof, M. van der Vlugt, *Hecke operators and the weight distributions of certain codes*, Journal of Combinatorial Theory 57, 1991, pp. 163 – 186.
- Wat35. G. N. Watson, *Generating functions of class numbers*, Compositio Mathematica, tome 1, 1935, pp. 39 – 68.
- Wei73. P. Weinberger, *Exponents of the class groups of complex quadratic fields*, Acta Arithmetica 22, 1973, pp. 117 – 124.
- Wes14. Hungabee specification, <https://www.westgrid.ca/support/systems/Hungabee>, 2014.

UNIVERSITY OF WATERLOO, 200 UNIVERSITY AVE W, WATERLOO, ONTARIO, CANADA N2L 3G1
E-mail address: `amosunov@uwaterloo.ca`

UNIVERSITY OF CALGARY, 2500 UNIVERSITY DRIVE NW, CALGARY, ALBERTA, CANADA T2N 1N4
E-mail address: `jacobs@cpsc.ucalgary.ca`